Available online at www.sciencedirect.com
ScienceDirect
Computer Standards & Interfaces 31 (2009) 219–226
COMPUTER STANDARDS & INTERFACES
www.elsevier.com/locate/csi

The anatomy study of server-initial agreement for general hierarchy wired/wireless networks

Chien-Fu Cheng [a], Shu-Ching Wang [b,*], Tyne Liang [a]

[a] Department of Computer Science, National Chiao Tung University, 1001, Ta Hsueh Rd., Hsinchu, 300, Taiwan, ROC
[b] Department of Information Management, Chaoyang University of Technology, 168 Gifeng E. Rd., Wufeng, Taichung County, 413, Taiwan, ROC

Received 11 July 2006; received in revised form 30 November 2007; accepted 20 December 2007
Available online 4 January 2008

Abstract

The Byzantine Agreement (BA) plays a key role in fault-tolerant distributed system design. A number of solutions to the BA problem based on various network model assumptions have been proposed. However, most existing BA protocols are designed for pure wired or pure wireless networks. In practice, most current networks are combined wired and wireless environments. In this paper, we extend the BA problem over a combined wired/wireless network, consisting of both powerful computing stationary processor and low-power mobile processor. The communication overhead of BA protocol is inherently large and secure group communications are important. The protocols proposed in this paper use the hierarchical model concept to reduce the communication overhead and provide secure group communications well suited for combined wired/wireless networks.

Keywords: Byzantine agreement protocol; Distributed computing; Fault-tolerance; Hierarchical model; Secure group communication

1. Introduction

The Byzantine Agreement (BA) problem introduced by Pease, Shostak and Lamport [9] is one of the fundamental problems in distributed computing. Assuming a distributed system with n ($n \geq 4$) processors, of which at most $n/3$ processors may be fallible, the source processor sends its initial value to the other processors in the beginning. All processors (without source processor) then exchange messages and a common agreement value is reached among all fault-free processors. More precisely, the BA problem is defined by the following two properties:

Agreement: All fault-free processors agree on the same common agreement value.
Validity: If the source processor is fault-free, the common agreement value should be the initial value of the source processor.

In previous results, the BA problem was solved in many network models with various fallible component assumptions, such as fully connected [9,15], generalized connected [12], general [7,11], multicasting [13] and mobile ad-hoc networks [14].

In [9], the network topology is fully connected network, and the fallible component assumption involves only malicious faulty processors. Detailed descriptions of faulty components are shown in Section 2. In [15], the network topology is also fully connected network, but the fallible component assumptions are malicious faulty processors and malicious faulty communication links. In [12], the assumption in a generalized connected network is that all network processors are partitioned into groups. Each group has the same number of processors, with the network topology fully connected. The fallible components include malicious faulty processors and malicious faulty communication links.

In the general network [7,11], the network topology may not be fully connected, the fallible components are dormant/malicious faulty processors, and dormant/malicious faulty communication links. In the multicasting network [13], processors are partitioned into groups, with each group having different numbers of processors, the network topology may not

* Corresponding author. Tel.: +886 4 23323000x4218; fax: +886 4 23742337.
E-mail address: scwang@cyut.edu.tw (S.-C. Wang).

0920-5489/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.csi.2007.12.004

be fully connected and fallible components are dormant/malicious faulty processors and dormant/malicious faulty communication links. In [14], the mobile ad-hoc network assumption is that each processor has mobility. The fallible components are malicious faulty processors and malicious faulty communication links. Many graceful BA protocols have been proposed according to the different network model assumptions. BA protocols have been proposed to ensure the reliability and fault-tolerance in different network models. Table 1 shows a comparison of various protocols over different network models.

The above-mentioned network models can be classified into two categories based on their mobility features. The two types are wired and wireless networks. A wired network consists of a hard-wired backbone and powerful computing processors. Therefore, the bandwidth speed, computation ability and reliability of wired networks are generally much better than those of wireless networks. A wireless network consists of mobile processors without a fixed infrastructure. Therefore, the wireless networks are very attractive for tactical communications in the military, law enforcement, and conferences [5].

However, the processors in a wired network do not have mobility. The limited resources (e.g., bandwidth and limited power) make the computation ability of mobile processors weaker than that of stationary processors. Therefore, most network environments today are combined wired and wireless. The combined wired/wireless networks have the advantages of both wired (e.g., powerful computation ability, high bandwidth, reliability, and so on.) and wireless networks (e.g., mobility, quick deployment, and so on).

Previous BA protocols for wired network [11–13,7,9,15] were not applicable in combined wired/wireless networks. Because, mobile processors may move away from the network during BA protocol execution and back to the network before ending the BA protocol, these mobile processors would not receive enough messages to reach a common agreement value. This situation violates the requirements of the BA problem in that "all fault-free processors" must agree on the same common agreement value. Previous BA protocol for wireless network [14] assumed that the communication between each processor is secure by using encryption technology. However, they did not explain how to create secure communication channel. Furthermore, the communication overhead of the BA protocol is inherently large because the BA protocol requires numerous rounds to exchange messages [4,6,9]. Previous BA protocols designed for flat architectures were not efficient because all messages must propagate globally throughout the network.

Table 1
The comparison of various protocols over different network models

	Network models				
	Fully connected network	Generalized connected network	General network	Multicasting network	Ad-hoc network
Lamport et al. [9]	V				
Yan et al. [15]	V				
Wang et al. [12]	V	V			
Meyer et al. [7]	V		V		
Siu et al. [11]	V		V		
Wang et al. [13]	V	V	V	V	
Wang et al. [14]	V		V		V

To solve the BA problem in the combined wired/wireless network, create secure communications between each processor and reduce the communication overhead, we propose a secure communication protocol and a hierarchical BA protocol. The proposed standard protocols called the SBAP (Server-initial Byzantine Agreement Protocol) and SGCP (Secure Group Communication Protocol). The SBAP uses the hierarchical model concept to reduce the communication overhead and provide secure group communications by SGCP.

The rest of this paper will proceed as follows. Section 2 introduces the conditions for the BA. Section 3 introduces the security technologies. The new approach is given in Section 4. An example of the proposed protocols is shown in Section 5. Section 6 gives the correctness and complexity of the SBAP. Conclusions are discussed in Section 7.

2. The conditions for Byzantine Agreement

The design and development of the BA protocol involves requirements that must be taken into account. There are the symptoms of a faulty processor, the symptoms of a faulty communication link, the system model, the number of message-exchange rounds, and the number of allowable faulty processors.

2.1. The symptoms of a faulty processor

A processor is fault-free if it follows the protocol specifications; otherwise, the processor said to be faulty. Faulty processor symptoms may include dormant fault (both crash and omission) or malicious fault (also called the Byzantine fault or the arbitrary fault) [7]. A crash fault occurs when a processor is broken. An omission fault takes place when a processor fails to transmit or receive a message on time. The most damaging fault type is the malicious fault (worst case), because the behavior of a malicious faulty processor is unpredictable and arbitrary. For example, a malicious faulty processor may work in coordination with other faulty processors to prevent other fault-free processors from reaching a common agreement value. If the BA problem can solve the worst case, the BA problem can also be solved under other fault type conditions. The proposed protocol will solve the BA problem with malicious faulty processors (worst case).

2.2. The symptoms of a faulty communication link

The symptoms of a faulty communication link can also be divided into two types. They are dormant fault (omission and delay) and malicious fault. In a synchronous system, each fault-free processor can detect messages from a dormant faulty communication link using the time-out mechanism or encryption technologies. Messages from a malicious faulty communication link can be detected by encryption technologies. In this study, the symptom of a faulty communication link is assumed malicious (worst case).

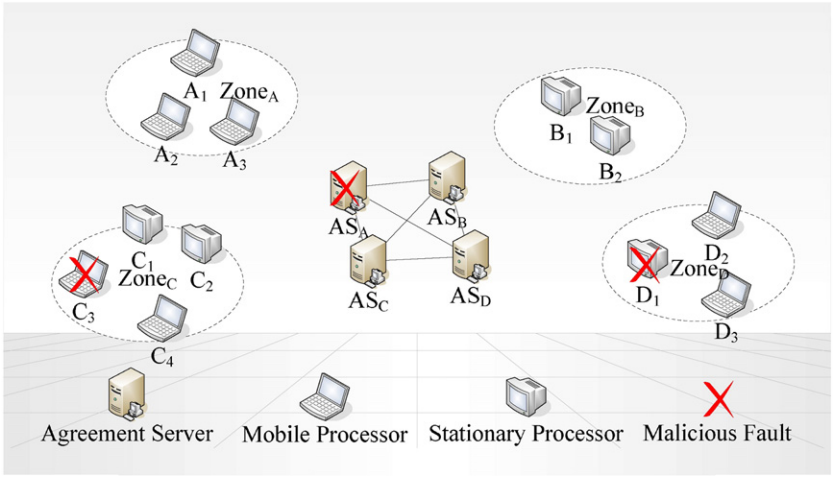

Fig. 1. An example of combined wired/wireless network.

2.3. The system model

Byzantine agreement protocols imply large communication overhead [4,6,9]. The previous network architectures from these results [6–9,1,4,11,14,15] were all flat architectures, with all processors carrying the same responsibility. BA protocols in flat architectures are not efficient because all messages must propagate globally throughout the network. To reduce the communication overhead, we used a hierarchical model concept. Our network model is a two-level combined wired/wireless computing environment consisting of a wired backbone and wireless cells that provide access to mobile processors.

Fig. 1 shows an example of the two-level combined wired/wireless network. There are sixteen processors in the network. There are four agreement–servers, five stationary processors and seven mobile processors. Each agreement–server manages a zone's processors. For example, agreement–server AS_A manages processor A_1, A_2 and A_3 in the zone A.

The assumptions and parameters of our system model are listed as follows:

- The underlying network is a two-level combined wired/wireless network.
- Processors include agreement–server, mobile processor and stationary processor.
- Agreement–server is a powerful and reliable computer with high bandwidth.
- Let N be the set of all processors in the network and $|N|=n$, where n is the number of processors in the underlying network.
- Let Z_N be the set of all agreement–servers in the network and $|Z_N|=z_n$, where z_n is the number of agreement–servers in the underlying network and $z_n \geq 4$.
- The underlying network is unreliable: messages may be dropped, reordered, inserted or duplicated by faulty processors or communication links.
- Each processor in the network can be identified uniquely.
- Let p_m be the maximum number of malicious faulty processors allowed.
- Let z_m be the maximum number of malicious faulty agreement–servers allowed.
- A processor does not know the faulty status of other processors in the underlying network.

2.4. The number of required rounds of message-exchange

In the BA protocol, we use term "round" to compute the amount of messages exchange. The term "round" denotes the interval of message exchange between any pair of processors [4,9]. Fischer and Lynch [4] indicated that $t+1$ ($t=\lfloor(n-1)/3\rfloor$) rounds are the minimum number of rounds required to get enough messages to achieve BA.

The network architecture by Fischer and Lynch [4] is flat architecture, but the network architecture in our system is a hierarchical architecture. In our protocol, only agreement–servers need to exchange messages in the Message Exchange Phase. Therefore, the number of required rounds of message-exchange is z_m+1 ($z_m=\lfloor(z_n-1)/3\rfloor$).

2.5. The constraints

In the BA problem, the number of faulty processors that can be allowed is determined by the total number of processors in the network. Pease, Shostak and Lamport [9] indicated the constraint of the BA problem is $n>3p_m$.

The network architecture of Pease, Shostak and Lamport [9] is flat architecture; so all processors need to exchange the

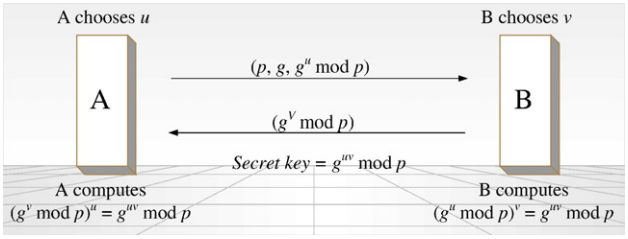

Fig. 2. Diffie–Hellman key exchange.

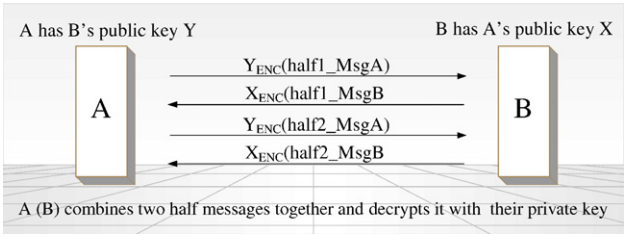

Fig. 3. Interlock protocol.

messages in the Message Exchange Phase. In our protocol, the network architecture is hierarchical, only agreement–servers need to exchange the messages in the Message Exchange Phase, so the constraint of our model is $z_n > 3z_m$.

3. Security technologies

Scure group communication is one of the important topics to provide secure communication in the network. In this section, we give a brief introduction of some security technologies that are used in our system as follows.

3.1. Public key infrastructure — asymmetric cryptographic algorithm

Public Key Infrastructure (PKI) acts as an important role for trust and authentication with each other in network. In this paper, we are not focus on how to build a reliable PKI. Therefore, we assume that the mutual authentication is finished. Some results have been proposed to solve this problem [10,16].

3.2. Diffie–Hellman key exchange

Diffie–Hellman key exchange [3] is a cryptographic protocol that allows two processors to agree on a secret key over an insecure communication channel. Once the shared secret key has been established, they can use it to encrypt their secret communication using conventional cryptographic methods. Fig. 2 shows the Diffie-Hellman key exchange procedure.

3.3. Interlock protocol

In cryptography, a man-in-the-middle attack (MITM) [10] is an attack in which an attacker is able to read, insert and modify at will, messages between two processors without either processor knowing that the link between them has been compromised. The attacker must be able to observe and intercept messages going between the two victims. Fig. 3 shows the MITM attack procedure.

Due to the MITM attack is particularly applicable to the original Diffie–Hellman key exchange protocol, when used without authentication. We use interlock protocol [10] to solve this problem. Fig. 4 shows the Interlock protocol procedure.

3.4. Advanced encryption standard — symmetric cryptographic algorithm

Advanced Encryption Standard (AES) also known as Rijndael [2] is a block cipher adopted as an encryption standard by the US government, is expected to be used worldwide, and analyses extensively as was the case with its predecessor the Data Encryption Standard (DES) [10].

4. Approach

In order to solve the BA problem over two-level combined wired/wireless network, we propose two standard protocols. They are Secure Group Communication Protocol (SGCP) and Server-initial Byzantine Agreement Protocol (SBAP).

4.1. Secure group communication protocol (SGCP)

The main goal of SGCP is to provide secure group communications between two processors in the two-level combined wired/wireless network and remove the influences from faulty communication links.

We know that mobile processor power is supplied using batteries. Because power saving is a serious topic with mobile processors. The asymmetric cryptographic algorithm, which needs a large amount of computation is not suited for mobile processors. The advantage of the symmetric cryptographic algorithm is that it is generally much faster than the asymmetric cryptographic algorithm. However, the disadvantage of the symmetric cryptographic algorithm is the requirement for a shared secret key with one copy at each end. Hence, maintaining secure during distribution is an important problem.

We combined the asymmetric cryptographic and symmetric cryptographic algorithms to obtain the advantages of both in this study. There are two phases in the SGCP, the Key Generation Phase and the Messages Transmission Phase.

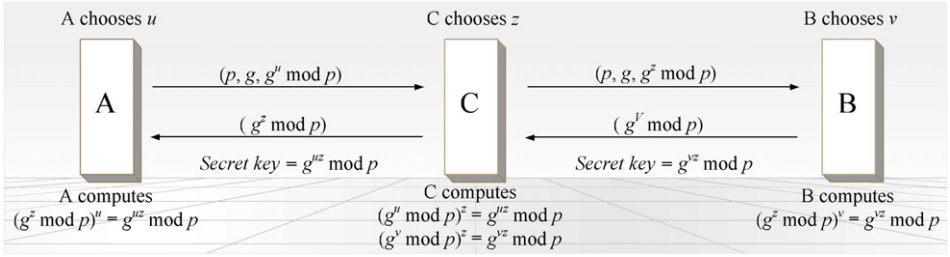

Fig. 4. Man-in-the-middle attack.

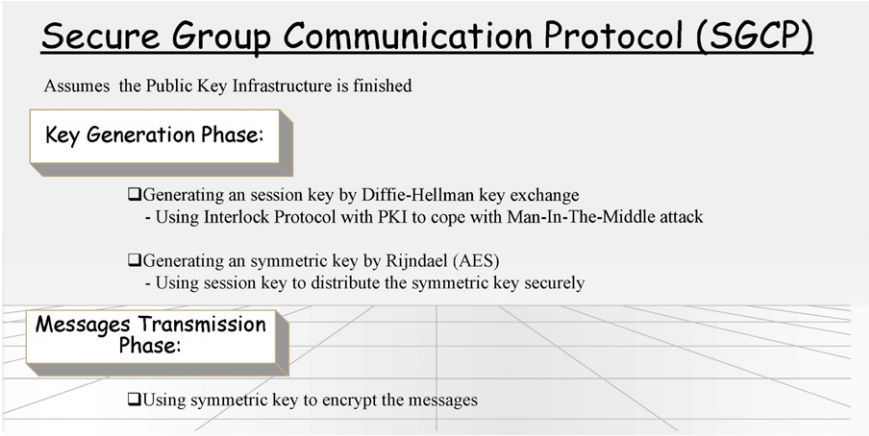

Fig. 5. Secure Group Communication Protocol (SGCP).

4.1.1. Key Generation Phase

The goal of the Key Generation Phase is used to generate a session key and symmetric key. The session key is generated using the Diffie–Hellman key exchange and interlock protocols with PKI to cope with the MITM attack. The symmetric key is generated using the Rijndael cipher algorithm (AES), which is a symmetric cryptographic algorithm. The symmetric key is distributed by session key.

4.1.2. Messages Transmission Phase

The goal of the Messages Transmission Phase is used to encrypt the messages. Since the symmetric key is generated by the Rijndael cipher algorithm (AES) is generally faster than the asymmetric cryptographic algorithm. Fig. 5 shows the SGCP. Fig. 6 shows secure group communications between two processors.

4.2. Server-initial Byzantine Agreement Protocol (SBAP)

To meet the characteristics of mobile environments in the BA problem, most of the communication and computation overhead must be fulfilled within in the agreement–servers. Therefore, only the agreement–server needs to exchange messages and compute the agreement value in our protocol. All messages in SBAP are encrypted by the symmetric key which is from SGCP to ensure the security. There are three phases in SBAP; they are Message Exchange Phase, Decision Making Phase and Agreement Distribution Phase. The protocol SBAP is shown in Fig. 7.

4.2.1. Message Exchange Phase

Each agreement–server computes the number of rounds required γ ($\gamma=z_m+1$, where $z_m=(z_n-1)/3$). At first round of Message Exchange Phase, only the source processor needs to encrypt its initial value to all other agreement–servers. Each agreement–server then stores the value from the source processor in the root of its mg-tree. At the $i\neq 1$ round of Message Exchange Phase, each agreement–server (without source processor) encrypts the value at level i-1 round in its mg-tree to all other agreement–servers. Each agreement–server then stores the value from other agreement–servers in the level i-th of its mg-tree.

4.2.2. Decision Making Phase

After Message Exchange Phase, each agreement–server deletes vertices with repeated names of mg-tree to avoid the repeated influence from faulty processors. Then, each agreement–server uses the VOTE_{mg} function on its mg-tree from leaf to root to obtain the agreement value.

4.2.3. Agreement Distribution Phase

Each agreement–server encrypts its agreement value to all processors in its zone. All fault-free processors (both stationary

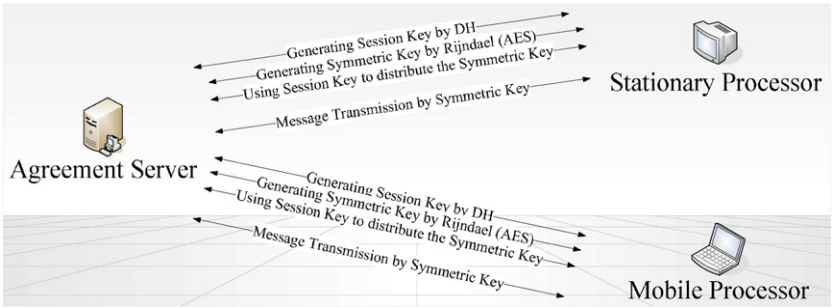

Fig. 6. The secure group communication.

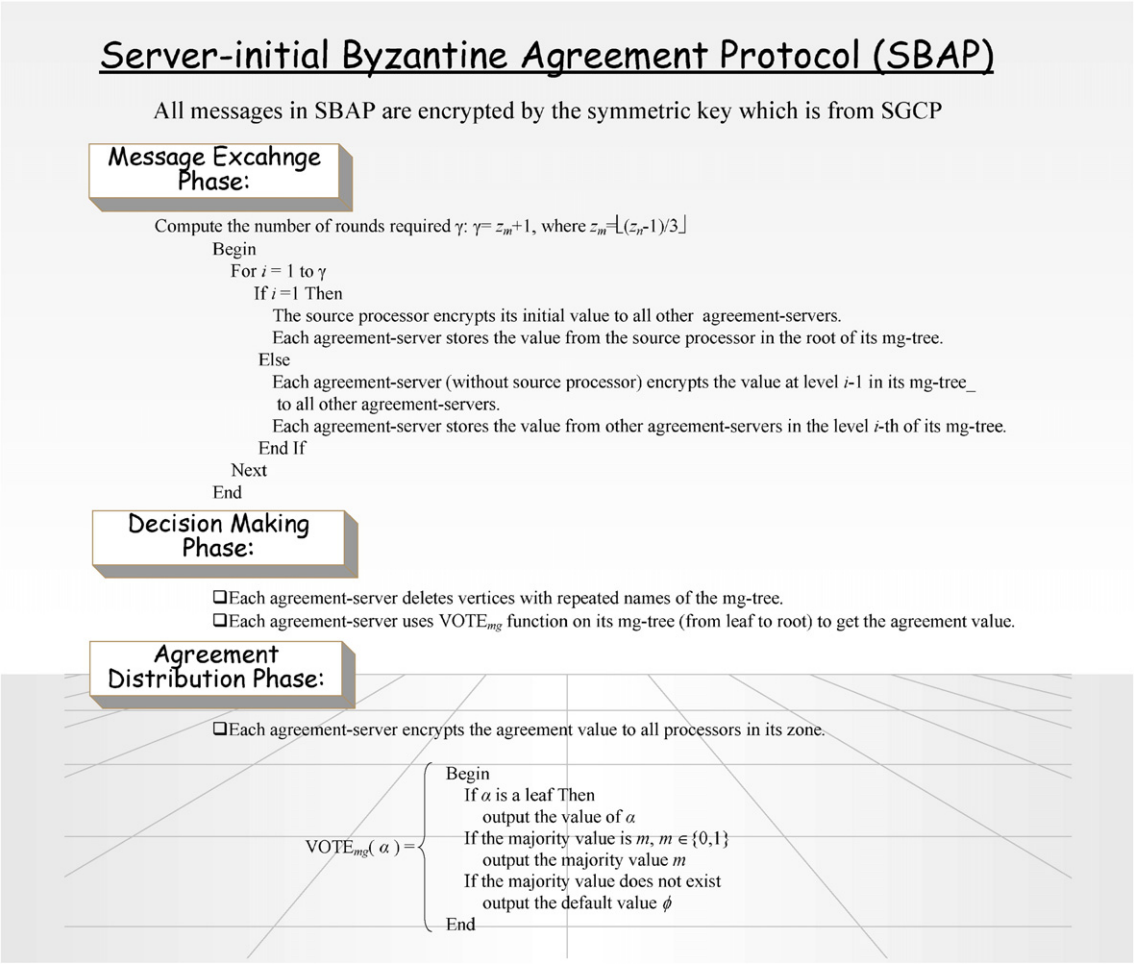

Fig. 7. The BA protocol Server-initial Byzantine Agreement Protocol (SBAP).

processors and mobile processors), which are managed by the fault-free agreement–server, can then reach a common agreement value. The value agreed upon by a faulty processor is ignored [9].

5. An example of executing protocol SBAP

In this section, we present a short synopsis of the SBAP execution protocol. A combined wire/wireless network is shown in Fig. 1. There are sixteen processors (including four agreement–servers, five stationary processors and seven mobile processors) falling into four zones. For example, there are three mobile processors (A_1, A_2 and A_3) in the zone A, and they are managed by agreement–server AS_A. The malicious faulty components are agreement–server AS_A, mobile processor C_3 and stationary processor D_1.

The source processor is the most important in the BA protocol. If the source processor has a malicious fault, it may send different initial values to different processors in the first round of Message Exchange Phase. Therefore, the worst case BA problem is that the source processor has a malicious fault. If the BA protocol can solve the worst case, the BA problem can be solved in other cases. Hence, we suppose that the agreement–server AS_A is the source processor. To reach a common agreement value among all fault-free component in our example, the SBAP needs 2 (⌊(4-1)/3⌋+1) message-exchange rounds.

In the first round of Message Exchange Phase, the source processor AS_A encrypts its initial value to all other agreement–servers in the network. Agreement–servers AS_B, AS_C, and AS_D then store the 10 value from the source processor AS_A in the root of their mg-trees, as shown in Fig. 8. In the second round of Message Exchange Phase, each agreement–server (without source processor) encrypts the value at the root in its mg-tree to all other agreement–servers. The 2-level mg-tree of agreement–server AS_B in the second round of Message Exchange Phase is shown in Fig. 9. In the Decision Making Phase, each agreement–server deletes the vertices with repeated mg-tree names to avoid the repeated influence from faulty processors. In our example, there is no vertex with a repeated name. The VOTE$_{mg}$ function is then used on its mg-tree from leaf to root to get the agreement value. For example, agreement–server ASB computes VOTE (A)=(0,1,1)=1 (VOTE(A)=(VOTE(AB), VOTE(AC), VOTE (AD))). An agreement value 1 is obtained. In the Agreement Distribution Phase, each agreement–server encrypts its agreement value to all processors in its zone. Therefore, agreement–server ASB encrypts its agreement value 1 to processor B_1 and processor B_2 the zone B.

6. The correctness and complexity of SBAP

If the value stored in vertex α of each fault-free agreement–server's mg-tree is identical, then the vertex α is called common [1]. When each fault-free agreement–server's has the common initial value from the source agreement–server in the root of its mg-tree, then an agreement is reached because the root is common. Thus, the constraints, (Agreement) and (Validity), can be rewritten as:

(Agreement'): Root value is common.
(Validity'): VOTE(α)=initial value of source agreement–server, for each fault-free agreement–server, if the source processor is fault-free.

To prove that a vertex is common, the term common frontier [1] is defined as follows: When every root-to-leaf path of the mg-tree contains a common vertex, the collection of the common vertices forms a common frontier. In other words, every fault-free agreement–server has the same messages collected in the common frontier if a common frontier does exist in a fault-free agreement–server's mg-tree. Subsequently, using the same function VOTEmg to compute the root value of the tree structure, every fault-free agreement–server can compute the same root value because the same input (the same collected messages in the common frontier) and the same computing function will produce the same output (the root value).

Lemma 1. All correct vertices of an mg-tree are common.

Proof. In the Decision Making Phase, all vertices with repeated names are deleted in an mg-tree. At level z_m+1 or above, the correct vertex α has at least $2z_m+1$ children, out of which at least z_m+1 children are correct. The true values of these z_m+1 correct vertices are common, and the majority of the vertex values α are common. The correct vertex α is common in the mg-tree if the level of α is less then z_m+1. Consequently, all correct vertices of the mg-tree are common. □

Lemma 2. The common frontier exists in the mg-tree.

Proof. There are z_m+1 vertices along each root-to-leaf path of an mg-tree in which the root is labeled by the source name, and the others are labeled by a sequence of agreement–server id. Since at most z_m agreement–server can fail, there is at least one correct vertex along each root-to-leaf path of the mg-tree. Using Lemma 1, the correct vertex is common and the common frontier exists in each fault-free agreement–server's mg-tree. □

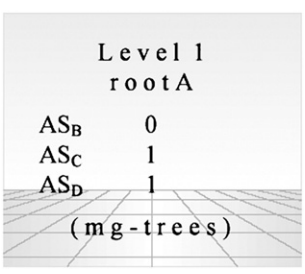

Fig. 8. The mg-trees of each agreement–server in the first round of Message Exchange Phase.

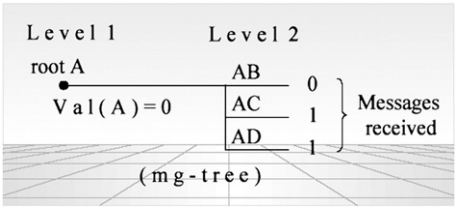

Fig. 9. The 2-level mg-tree of agreement–server AS_B.

Lemma 3. Let α be a vertex, α is common if there is a common frontier in the sub-tree rooted at α.

Proof. If the height of α is 0 and the common frontier (α itself) exists, α is common. If the height of α is γ, the children of α are all in common under the induction hypothesis with the height of the children being $\gamma-1$. □

Corollary 1. The root is common if the common frontier exists in the mg-tree.

Theorem 1. The root of a fault-free agreement–server's mg-tree is common.

Proof. Using Lemmas 1, 2, 3 and Corollary 1, the theorem is proved. □

Theorem 2. Protocol SBAP solves the BA problem in a two-level combined wired/wireless network.

Proof. To prove this theorem, SBAP must meet the constraints (Agreement') and (Validity')

(Agreement'): Root value is common. By Theorem 1, (Agreement') is satisfied
(Validity'): VOTE(α)=v for all fault-free agreement–servers, if the initial value of the source agreement–server is v_s say $v=v_s$.

Most agreement–servers are fault-free. The value of the correct vertices for all of the fault-free agreement–servers' mg-trees is v. Therefore, each correct vertex of the mg-tree is common (Lemma 1), and its true value is v. Using Theorem 1, this root is common. The computed value VOTE(α)=v is stored in the root for all the fault-free agreement–server. Therefore, (Validity') is satisfied. □

Theorem 3. SBAP requires z_m+1 rounds in the "Message Exchange Phase" to solve the BA problem in a two-level combined wired/wireless network, and z_m+1 ($z_m=\lfloor(z_n-1)/3\rfloor$) is the minimum number of rounds in the "Message Exchange Phase".

Proof. The "Message Exchange Phase" is a time consuming phase. Fischer and Lynch [4] indicated that $t+1$ ($t=\lfloor(n-1)/3\rfloor$) rounds are the minimum number of rounds required to get enough messages to achieve BA. The network architecture of Fischer and Lynch [4] is flat architecture, but the network architecture of our system is hierarchical architecture. In our protocol, only agreement–servers need to exchange the messages in the Message Exchange Phase, so the number of required rounds of message-exchange is z_m+1 ($z_m=\lfloor(z_n-1)/3\rfloor$). Thus, SBAP requires z_m+1 rounds, and this number is the minimum. □

7. Conclusion

Combined wired/wireless networks have become popular because they have the advantages of both wired network (e.g., powerful computation ability, high bandwidth, reliability and so on.) and wireless network (e.g., mobility, quick deployment and so on). Previous BA protocols [1,4,6–15] were not applicable for combined wired/wireless networks. In this paper, we revisit the BA problem over a combined wired/wireless network with malicious faulty components and use a hierarchical architecture to reduce the communication overhead.

Base on the preceding discussion, the protocol SBAP and SGCP have the following features:

- ➢ SGCP provides a secure group communication in the combined wired/wireless network.
  - SGCP combines asymmetric and symmetric cryptographic algorithms to get the advantages of both.
  - SGCP is well suited for the combined wired/wireless network.
- ➢ Most of the communication and computation overhead are fulfilled within in agreement–servers
  - To meet the characteristics of mobile environments, most of the communication and computation overhead must be fulfilled within in the agreement–servers.
- ➢ SBAP can reduce the number of message-exchange rounds
  - SBAP uses the hierarchical model concept to reduce the number of message-exchange rounds.
- ➢ SBAP can reach a common agreement with malicious faulty components (processors and communication links) over two-level combined wired/wireless networks.
  - By Theorem 2.
- ➢ The number of message-exchange rounds for SBAP is the minimum.
  - By Theorem 3.

In summary, the SGCP protocol can provide secure group communication and the SBAP protocol can solve the BA problem with malicious faulty processors and malicious faulty communication links over two-level combined wired/wireless network. Moreover, SBAP uses only z_m+1 ($z_m=\lfloor(z_n-1)/3\rfloor$) rounds of message-exchange and can tolerate the maximum number of allowable malicious faulty processors and malicious faulty communication links to make all fault-free processors (both stationary processors and mobile processors), which are managed by a fault-free agreement–server, reach the same common agreement value.

References

[1] M. Barborak, M. Malek, A. Dahbura, The consensus problem in fault-tolerant computing, ACM Computing Surveys 25 (1993) 171–220.
[2] J. Daemen, V. Rijmen, The Rijndael block cipher, AES Document Version2, 1999.
[3] W. Diffie, M.E. Hellman, New directions in cryptography, IEEE Transactions on Information Theory 22 (1976) 644–654.
[4] M. Fischer, N. Lynch, A lower bound for the assure interactive consistency, Information Processing Letters 14 (4) (1982) 183–186.
[5] X. Hong, K. Xu, M. Gerla, Scalable routing protocols for mobile ad hoc networks, IEEE Networks 16 (4) (2002) 11–21.
[6] J. Martin, L. Alvisis, Fast byzantine consensus, Proceeding of the 2005 International Conference on Dependable Systems and Networks, 2005.
[7] F.J. Meyer, D.K. Pradhan, Consensus with dual failure modes, IEEE Transaction on Parallel and Distributed Systems 2 (2) (1991) 214–222.
[8] M. Okum, Agreement among unacquainted Byzantine generals, Lecture Note on Computer Sciences 3724 (2005) 499–500.
[9] M. Pease, R. Shostak, L. Lamport, Reaching agreement in the presence of faults, Journal of ACM 27 (2) (1980) 228–234.
[10] B. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, John Whily & Sons, Inc., 1994
[11] H.S. Siu, Y.H. Chin, W.P. Yang, Byzantine agreement in the presence of mixed faults on processors and links, IEEE Transaction on Parallel and Distributed System 9 (4) (1998) 335–345.
[12] S.C. Wang, Y.H. Chin, K.Q. Yan, Byzantine agreement in a generalized connected network model, IEEE Transaction on Parallel and Distributed System 6 (4) (1995) 420–427.
[13] S.C. Wang, K.Q. Yan, C.F. Cheng, Achieving high efficient Byzantine Agreement with dual components failure mode on a multicasting network, Proceeding of the 9th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2002), 2002, pp. 577–582.
[14] S.C. Wang, W.P. Yang, C.F. Cheng, Byzantine Agreement on mobile ad-hoc network, Proceeding of the IEEE International Conference on Networking, Sensing and Control (ICNSC 2004), 2004, pp. 52–57.
[15] K.Q. Yan, Y.H. Chin, S.C. Wang, Optimal agreement protocol in malicious faulty processors and faulty links, IEEE Transaction on Knowledge and Data Engineering 4 (3) (1992) 266–280.
[16] S. Yi, R. Kravets, Key management for heterogeneous ad hoc wireless networks, Proceedings of the 10th IEEE International Conference on Network Protocols (ICNP2002), 2002, pp. 202–203.

C.F. Cheng received the B.S. degree in Information Management and M.S. degree in Computer Science and Information Engineering from Chaoyang University of Technology, Taiwan.
Currently, he is a lecturer with Department of Computer Science and Information Engineering of Yuanpei University, Taiwan. He is pursuing his Ph.D. in Computer Science from National Chiao-Tung University, Taiwan. His current research interests include distributed data processing, fault tolerant computing, and mobile computing.

S.C. Wang received the B.S. degree in Computer Science from Feng-Chia University, the M.S. degree in Electrical Engineering from National Chen-Kung University, and Ph.D. degree in Information Engineering from National Chiao-Tung University, Taiwan.
Currently, she is a Professor with the Department of Information Management, Chaoyang University of Technology, Taichung County, Taiwan. Her current research interests include grid computing, distributed data processing, parallel processing, and algorithm analysis and design.

T. Liang received her Ph.D. degree in Computer Science and Information Engineering from National Chiao-Tung University, Taiwan.
Currently, she is an associate professor with Department of Computer Science, National Chiao-Tung University. Her research interests include information retrieval, natural language processing and interconnection network.